In [2]:
```
import numpy as np
import pandas as pd

import nltk
from nltk.corpus import stopwords
import string
from wordcloud import WordCloud

import seaborn as sns

import matplotlib.pyplot as plt
```

In [3]:
```
#reading the data

df = pd.read_csv('/home/dara/Text_Analytics/Resume_Data.csv')
df['Cleaned_Resume'] = ''
df.head()
```

Out[3]:

	Category	Resume	Cleaned_Resume
0	Data Science	Skills * Programming Languages: Python (pandas...	
1	Data Science	Education Details \r\nMay 2013 to May 2017 B.E...	
2	Data Science	Areas of Interest Deep Learning, Control Syste...	
3	Data Science	Skills â ¢ R â ¢ Python â ¢ SAP HANA â ¢ Table...	
4	Data Science	Education Details \r\n MCA YMCAUST, Faridab...	

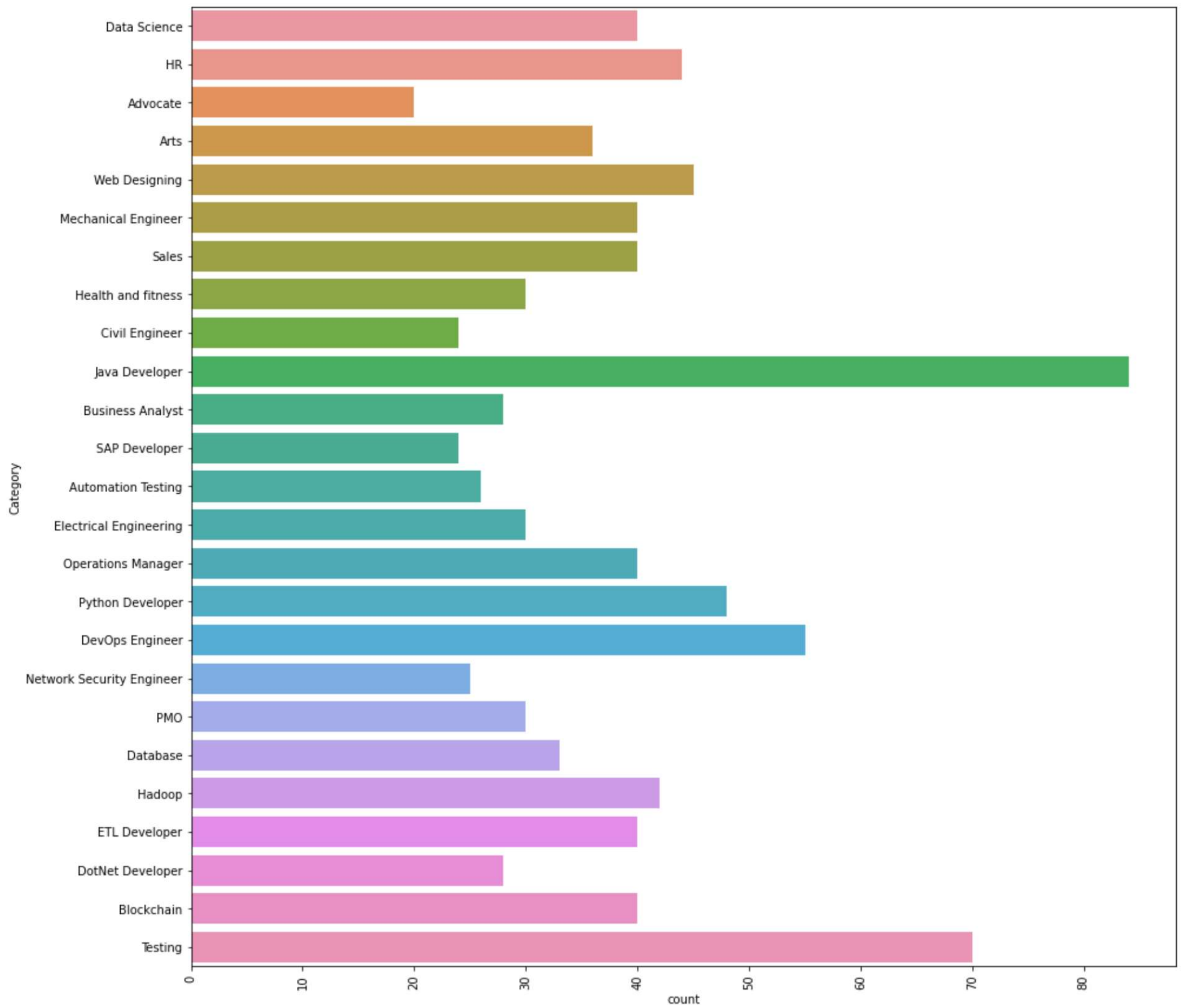
Cleaned_Resume is created to keep the clean text.

In [4]:
```
print ("Resume Categories")
print (df['Category'].value_counts())
```

```
Resume Categories
Java Developer               84
Testing                      70
DevOps Engineer              55
Python Developer             48
Web Designing                45
HR                           44
Hadoop                       42
Blockchain                   40
ETL Developer                40
Operations Manager           40
Data Science                 40
Sales                        40
Mechanical Engineer          40
Arts                         36
Database                     33
Electrical Engineering       30
Health and fitness           30
PMO                          30
Business Analyst             28
DotNet Developer             28
Automation Testing           26
Network Security Engineer    25
SAP Developer                24
Civil Engineer               24
Advocate                     20
Name: Category, dtype: int64
```

In [5]:
```
plt.figure(figsize=(15,15))
plt.xticks(rotation=90)
sns.countplot(y="Category", data=df)
```

Out[5]: <AxesSubplot:xlabel='count', ylabel='Category'>



In [6]:
```
df["Resume"][2]
```

Out[6]:
```
'Areas of Interest Deep Learning, Control System Design, Programming in-Python,
Electric Machinery, Web Development, Analytics Technical Activities q Hindustan
Aeronautics Limited, Bangalore - For 4 weeks under the guidance of Mr. Satish,
Senior Engineer in the hangar of Mirage 2000 fighter aircraft Technical Skills
Programming Matlab, Python and Java, LabView, Python WebFrameWork-Django, Flas
k, LTSPICE-intermediate Languages and and MIPOWER-intermediate, Github (GitBas
h), Jupyter Notebook, Xampp, MySQL-Basics, Python Software Packages Interpreter
s-Anaconda, Python2, Python3, Pycharm, Java IDE-Eclipse Operating Systems Windo
ws, Ubuntu, Debian-Kali Linux Education Details \r\nJanuary 2019 B.Tech. Electr
ical and Electronics Engineering  Manipal Institute of Technology\r\nJanuary 20
15    DEEKSHA CENTER\r\nJanuary 2013    Little Flower Public School\r\nAugust 2
000    Manipal Academy of Higher\r\nDATA SCIENCE \r\n\r\nDATA SCIENCE AND ELECT
RICAL ENTHUSIAST\r\nSkill Details \r\nData Analysis- Exprience - Less than 1 ye
ar months\r\nexcel- Exprience - Less than 1 year months\r\nMachine Learning- Ex
prience - Less than 1 year months\r\nmathematics- Exprience - Less than 1 year
months\r\nPython- Exprience - Less than 1 year months\r\nMatlab- Exprience - Le
ss than 1 year months\r\nElectrical Engineering- Exprience - Less than 1 year m
onths\r\nSql- Exprience - Less than 1 year monthsCompany Details \r\ncompany -
THEMATHCOMPANY\r\ndescription - I am currently working with a Casino based oper
ator(name not to be disclosed) in Macau.I need to segment the customers who vis
it their property based on the value the patrons bring into the company.Basical
ly prove that the segmentation can be done in much better way than the current
system which they have with proper numbers to back it up.Henceforth they can im
plement target marketing strategy to attract their customers who add value to t
he business.'
```

As we can see the text needs a lot of processing. This is not suitable for analyzing

In [7]:
```
#We now have to clean the resume text.
#re--lets you check if a particular string matches a given regular expression
import re
def cleanResume(resumeText):
    resumeText = re.sub('http\S+\s*', ' ', resumeText)  # remove URLs
    resumeText = re.sub('RT|cc', ' ', resumeText)  # remove RT and cc
    resumeText = re.sub('#\S+', '', resumeText)  # remove hashtags
    resumeText = re.sub('@\S+', '  ', resumeText)  # remove mentions
    resumeText = re.sub('[%s]' % re.escape("""!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~"""
    resumeText = re.sub(r'[^\x00-\x7f]',r' ', resumeText)
    resumeText = re.sub('\s+', ' ', resumeText)  # remove extra whitespace
    return resumeText

df['Cleaned_Resume'] = df.Resume.apply(lambda x: cleanResume(x))
```

In [8]:
```
df.head()
```

Out[8]:

	Category	Resume	Cleaned_Resume
0	Data Science	Skills * Programming Languages: Python (pandas...	Skills Programming Languages Python pandas num...
1	Data Science	Education Details \r\nMay 2013 to May 2017 B.E...	Education Details May 2013 to May 2017 B E UIT...
2	Data Science	Areas of Interest Deep Learning, Control Syste...	Areas of Interest Deep Learning Control System...
3	Data Science	Skills â ¢ R â ¢ Python â ¢ SAP HANA â ¢ Table...	Skills R Python SAP HANA Tableau SAP HANA SQL ...
4	Data Science	Education Details \r\n MCA YMCAUST, Faridab...	Education Details MCA YMCAUST Faridabad Haryan...

Now we see that the text is clean.

In [9]:
```
len(df)
```

Out[9]: 962

In [10]:
```
#getting the entire Cleaned_Resume as single text.

corpus=" "

for i in range(0,962):
    corpus= corpus+ df["Cleaned_Resume"][i]
```

In [11]:
```
corpus[1000:2500]
```

Out[11]:
```
'review process and run analytics and generate reports Core member of a team he
lped in developing automated review platform tool from scratch for assisting E
discovery domain this tool implements predictive coding and topic modelling by
automating reviews resulting in reduced labor costs and time spent during the l
awyers review Understand the end to end flow of the solution doing research and
development for classification models predictive analysis and mining of the inf
ormation present in text data Worked on analyzing the outputs and precision mon
itoring for the entire tool TAR assists in predictive coding topic modelling fr
om the evidence by following EY standards Developed the classifier models in or
der to identify red flags and fraud related issues Tools Technologies Python sc
ikit learn tfidf word2vec doc2vec cosine similarity Na ve Bayes LDA NMF for top
ic modelling Vader and text blob for sentiment analysis Matplot lib Tableau das
hboard for reporting MULTIPLE DATA SCIENCE AND ANALYTIC PROJECTS USA CLIENTS TE
XT ANALYTICS MOTOR VEHICLE CUSTOMER REVIEW DATA Received customer feedback surv
ey data for past one year Performed sentiment Positive Negative Neutral and tim
e series analysis on customer comments across all 4 categories Created heat map
of terms by survey category based on frequency of words Extracted Positive and
Negative words across all the Survey categories and plotted Word cloud Created
customized tableau dashboards for effective reporting and visualizations CHAT'
```

As the text has now been cleaned and joined together and is ready for document preprocessing methods.

Tokenization

Tokenization is the process of breaking raw text into small units. Here, we convert the entire text into single words. Tokenization is important because it splits the data into small usable and easy-to-process units. These smaller units of text are called tokens. These tokens can help in understanding the context of the text and also in building the NLP models.

In [12]:
```
#Creating the tokenizer
tokenizer = nltk.tokenize.RegexpTokenizer('\w+')

#Tokenizing the text
tokens = tokenizer.tokenize(corpus)

len(tokens)
```

Out[12]: 411913

In [13]:
```
#now we shall make everything lowercase for uniformity
#to hold the new lower case words

words = []

#Looping through the tokens and make them lower case
for word in tokens:
    words.append(word.lower())
```

Here we have used word tokenization for our analyzing.

POS Tagging

POS Tagging is a popular Natural Language Processing process which refers to categorizing word in a text (corpus) in correspondance with a particular part of speech, depending on the definition of the word and it's context.

In [14]:
```
words1 = nltk.word_tokenize(corpus)
```

In [15]:
```
print(words1)
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

In [16]:
```
len(words1)
```

Out[16]:
```
411913
```

In [17]:
```
import nltk
nltk.download('averaged_perceptron_tagger')
nltk.pos_tag(words1)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /home/dara/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

In [18]:
```
import nltk
nltk.download('tagsets')
nltk.help.brown_tagset()
```

```
(: opening parenthesis
    (
): closing parenthesis
    )
*: negator
    not n't
,: comma
    ,
--: dash
    --
.: sentence terminator
    . ? ; ! :
:: colon
    :
ABL: determiner/pronoun, pre-qualifier
    quite such rather
ABN: determiner/pronoun, pre-quantifier
    all half many nary
ABX: determiner/pronoun, double conjunction or pre-quantifier
```

In [19]:
```
nltk.help.upenn_tagset('NNP')
```

```
NNP: noun, proper, singular
    Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos
    Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin ODI Darryl CTCA
    Shannon A.K.C. Meltex Liverpool ...
```

Stop words removal

For analyzing text and NLP, stopwords are removed from the text, as they do not add much value and meaning to the text. Stopwords, if added would bring in a lot of unnecessary noise and be of no use to the analytics process. Also, the removal of stopwords reduces the amount of data we have to process, thus reducing the number of tokens and makes everything faster.

Examples of Stopwords in English: ‘nor’, ‘me’, ‘were’, ‘her’, ‘more’, ‘himself’, ‘this’.

In [20]:
```
#Stop words are generally the most common words in a language.
#English stop words from nltk.

stopwords = nltk.corpus.stopwords.words('english')

words_new = []

#Now we need to remove the stop words from the words variable
#Appending to words_new all words that are in words but not in stopwords

for word in words:
    if word not in stopwords:
        words_new.append(word)
```

In [21]:
```
len(words_new)
```

Out[21]:
```
318305
```

Stemming and Lemmatization

Stemming just removes the last few characters of a word, often leading incorrect meanings and spelling.

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item.

Lemmatization is similar to stemming but it brings context to the words. So it links words with similar meaning to one word.

Lemmatization is preferred over Stemming because lemmatization does morphological analysis of the words.

In [22]:
```
from nltk.stem import WordNetLemmatizer

wn = WordNetLemmatizer()

lem_words=[]

for word in words_new:
    word=wn.lemmatize(word)
    lem_words.append(word)
```

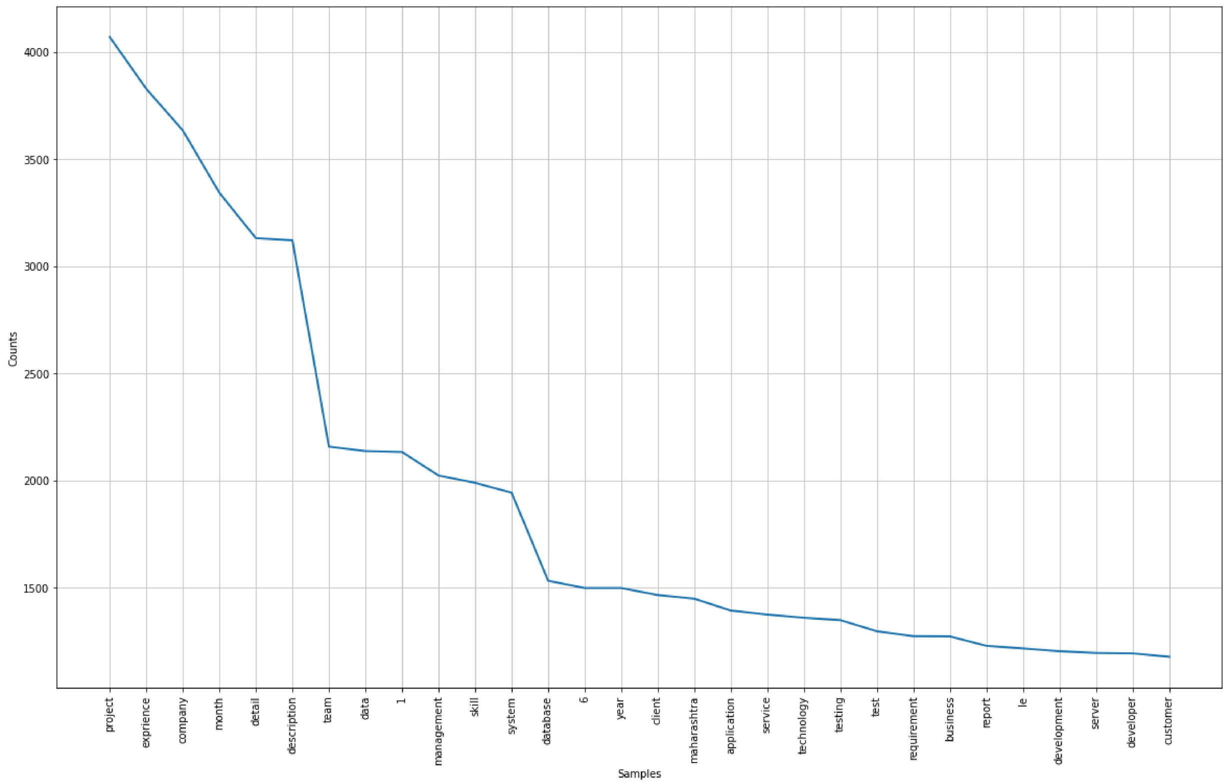
In [23]:
```
same=0
diff=0

for i in range(0,1832):
    if(lem_words[i]==words_new[i]):
        same=same+1
    elif(lem_words[i]!=words_new[i]):
        diff=diff+1

print('Number of words Lemmatized=', diff)
print('Number of words not Lemmatized=', same)
```

```
Number of words Lemmatized= 294
Number of words not Lemmatized= 1538
```

Now, with the Lemmatization done, we proceed to get the Frequency Distribution.

Frequency Distribution

In [27]:
```
#The frequency distribution of the words
freq_dist = nltk.FreqDist(lem_words)
#Frequency Distribution Plot
plt.subplots(figsize=(20,12))
freq_dist.plot(30)
```



Out[27]: <AxesSubplot:xlabel='Samples', ylabel='Counts'>

In [28]:
```
mostcommon = freq_dist.most_common(50)
mostcommon
```

Out[28]:
```
[('project', 4071),
 ('exprience', 3829),
 ('company', 3635),
 ('month', 3344),
 ('detail', 3132),
 ('description', 3122),
 ('team', 2159),
 ('data', 2138),
 ('1', 2134),
 ('management', 2024),
 ('skill', 1990),
 ('system', 1944),
 ('database', 1533),
 ('6', 1499),
 ('year', 1499),
 ('client', 1466),
 ('maharashtra', 1449),
 ('application', 1394),
 ('service', 1375),
 ('technology', 1360),
 ('testing', 1349),
 ('test', 1297),
 ('requirement', 1274),
 ('business', 1273),
 ('report', 1229),
 ('le', 1217),
 ('development', 1204),
 ('server', 1196),
 ('developer', 1194),
 ('customer', 1178),
 ('ltd', 1177),
 ('process', 1163),
 ('responsibility', 1137),
 ('using', 1124),
 ('sql', 1120),
 ('january', 1090),
 ('java', 1076),
 ('engineering', 1055),
 ('work', 1038),
 ('pune', 1026),
 ('role', 969),
 ('c', 951),
 ('user', 916),
 ('operation', 895),
 ('software', 886),
 ('pvt', 879),
 ('sale', 845),
 ('activity', 832),
 ('environment', 800),
 ('design', 786)]
```

We can have a look at the frequency distribution, words like project, company, management, team, etc are very common. Having a look at the entire frequency table will show which types of words are more used.

Recruiters can apply these analytics to understand the general profile of the applicants. Often screening and applicant selection are done on metrics gathered from Resume text.

WordCloud

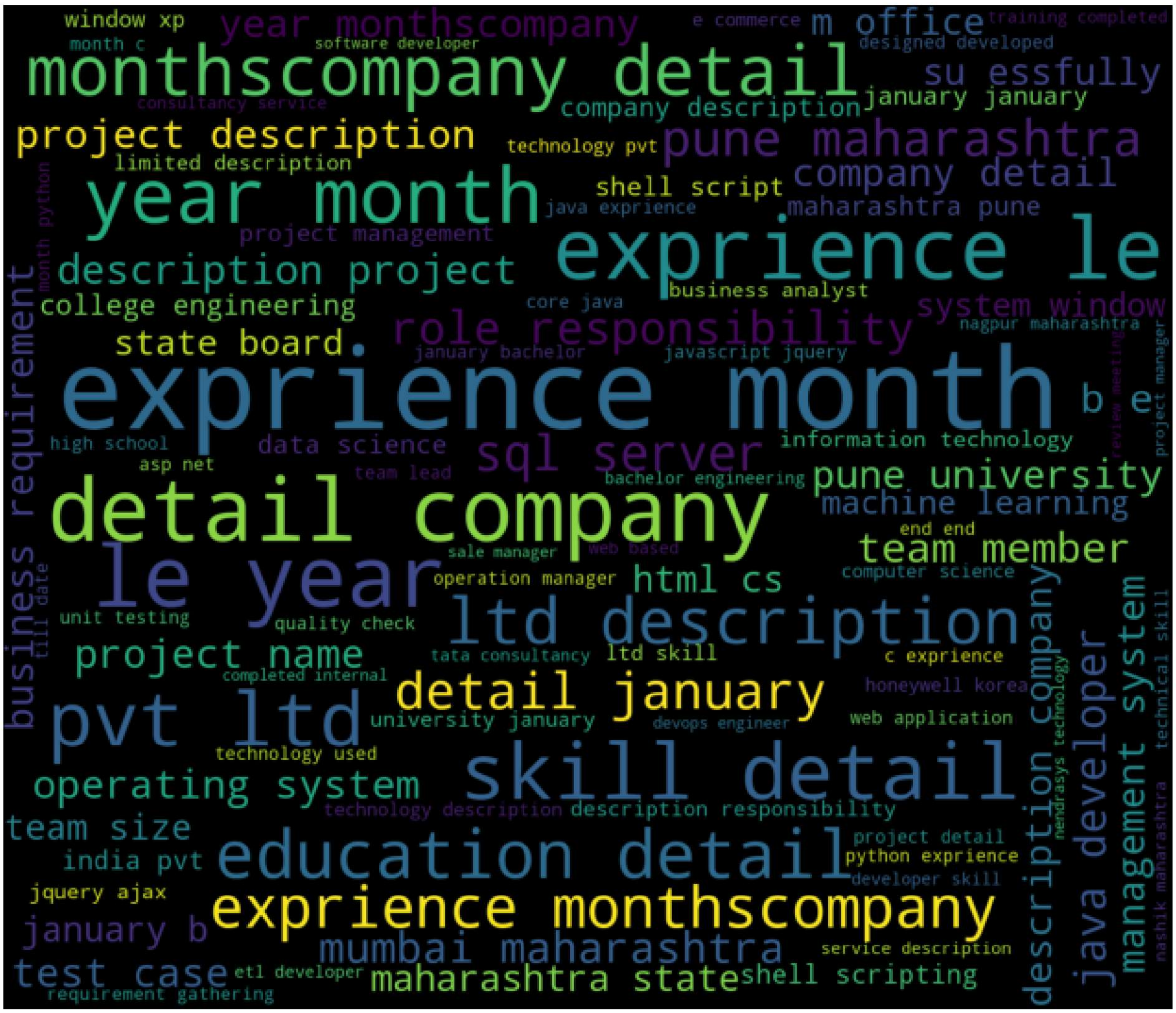
We have generated WordCloud for 200 words.

Size of the word is determined by their frequency.

In [25]:
```
#converting into string
res=' '.join([i for i in lem_words if not i.isdigit()])
```

In [26]:
```python
plt.subplots(figsize=(16,10))
wordcloud = WordCloud(
                          background_color='black',
                          max_words=100,
                          width=1400,
                          height=1200
                         ).generate(res)
plt.imshow(wordcloud)
plt.title('Resume Text WordCloud (100 Words)')
plt.axis('off')
plt.show()
```

Resume Text WordCloud (100 Words)



In [ ]: